MICHIGAN STATE UNIVERSITY

Project Plan

Investment Portfolio Construction

The Capstone Experience

Team Principal IPC

Sean Kennedy
Don Nakashima
John Parke
Yue Wang
Andrew Watson

Department of Computer Science and Engineering
Michigan State University

Spring 2020

From Students…
…to Professionals

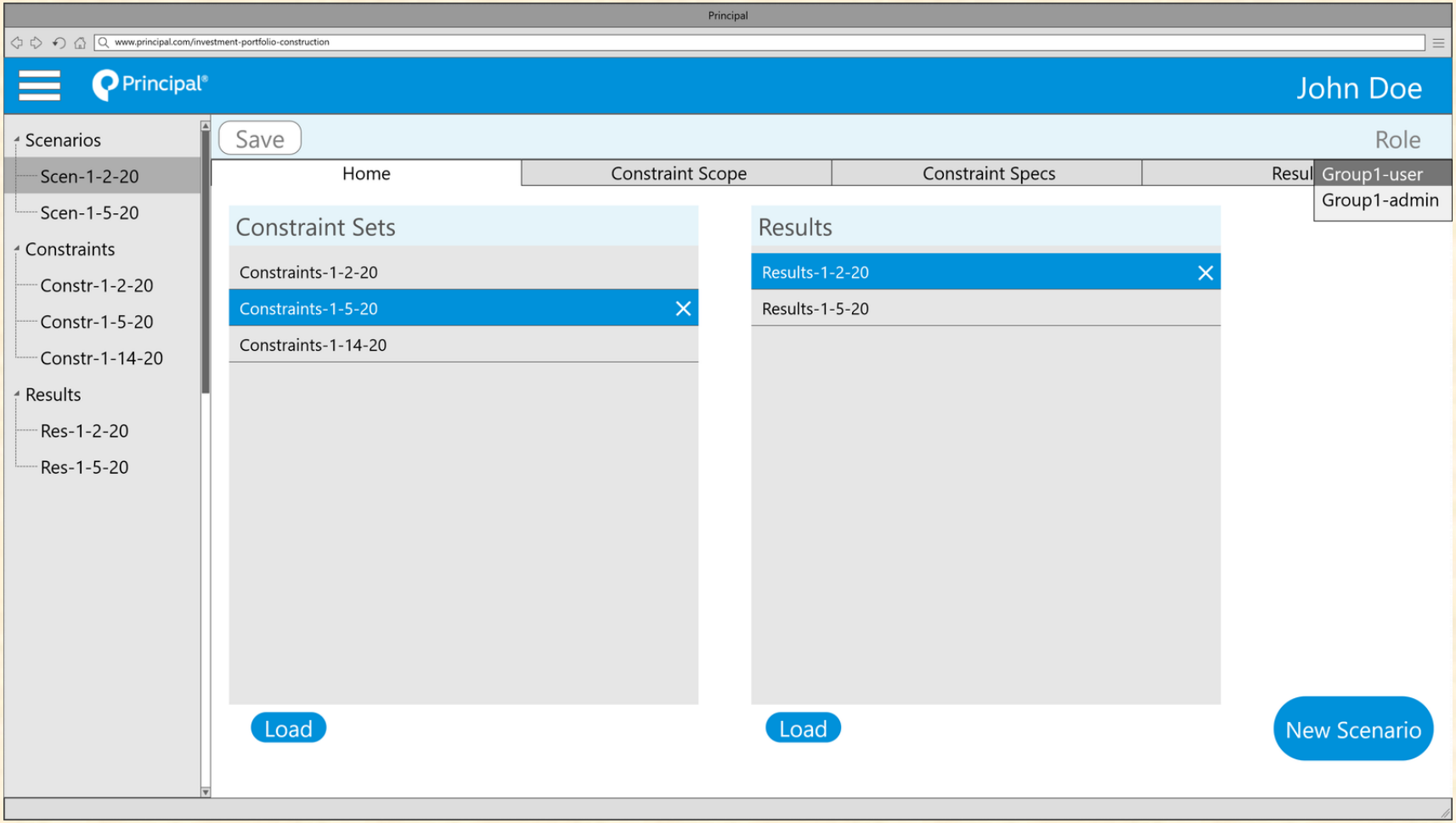
Functional Specifications

- Principal's current process involves hard-coding parameters to generate a single type of investment portfolio construction
- Our application provides a user interface for saving and loading optimization parameters and passing them to Principal's existing optimization engine
- The application assigns groups to its users, providing them with the ability to construct a variety of portfolio constructions based on custom-built portfolio-level and quantile-level constraints

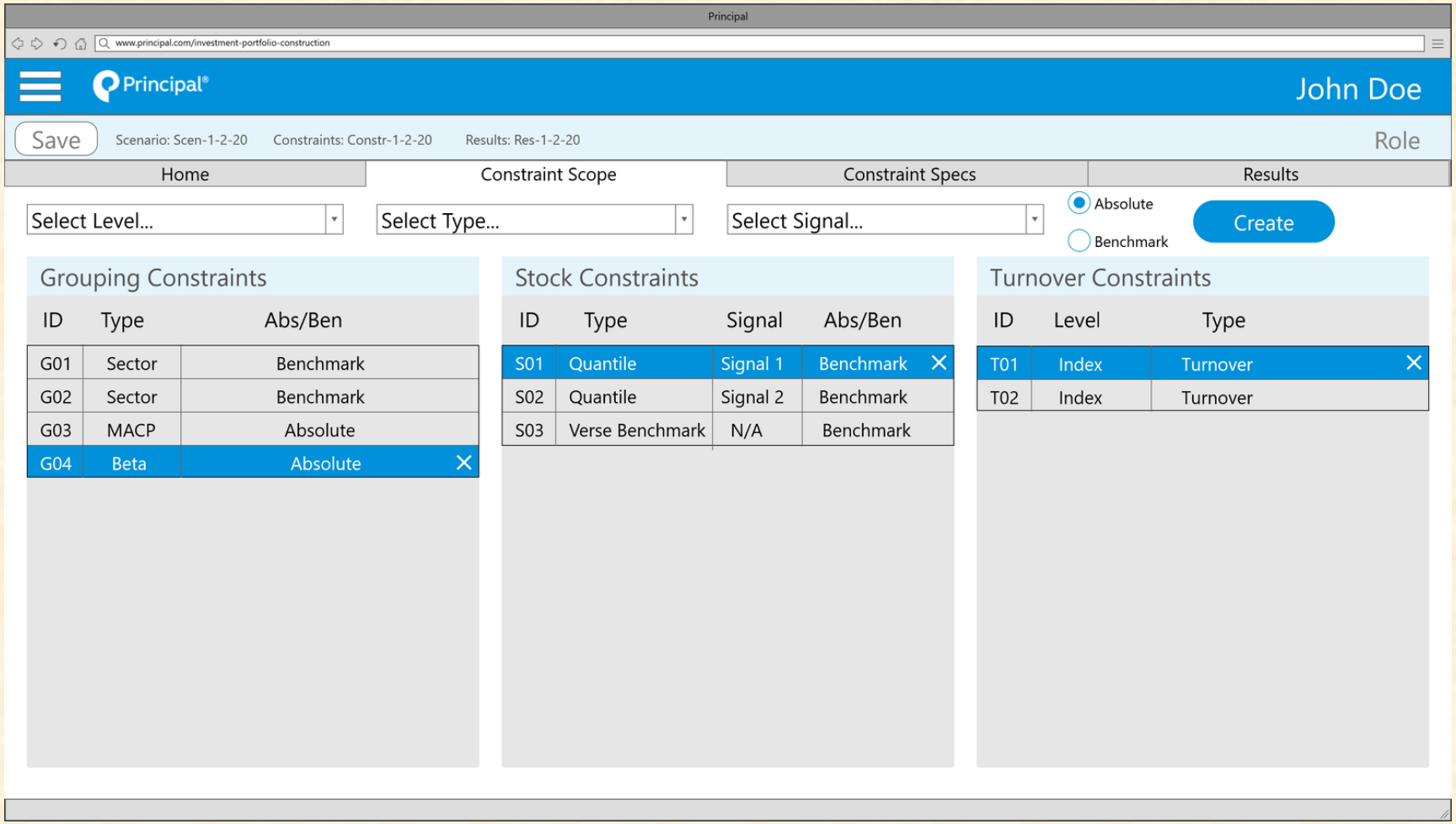
Design Specifications

- Landing Page: View any constraint sets or results saved for specific user/group
- Constraint Scoping Page: Create the basis for user constraints in a portfolio construction scenario
- Constraint Specification Page: Configure specific settings for each of the constraints defined on scoping page
- Results page: Displays the investment portfolio construction returned by the optimization engine

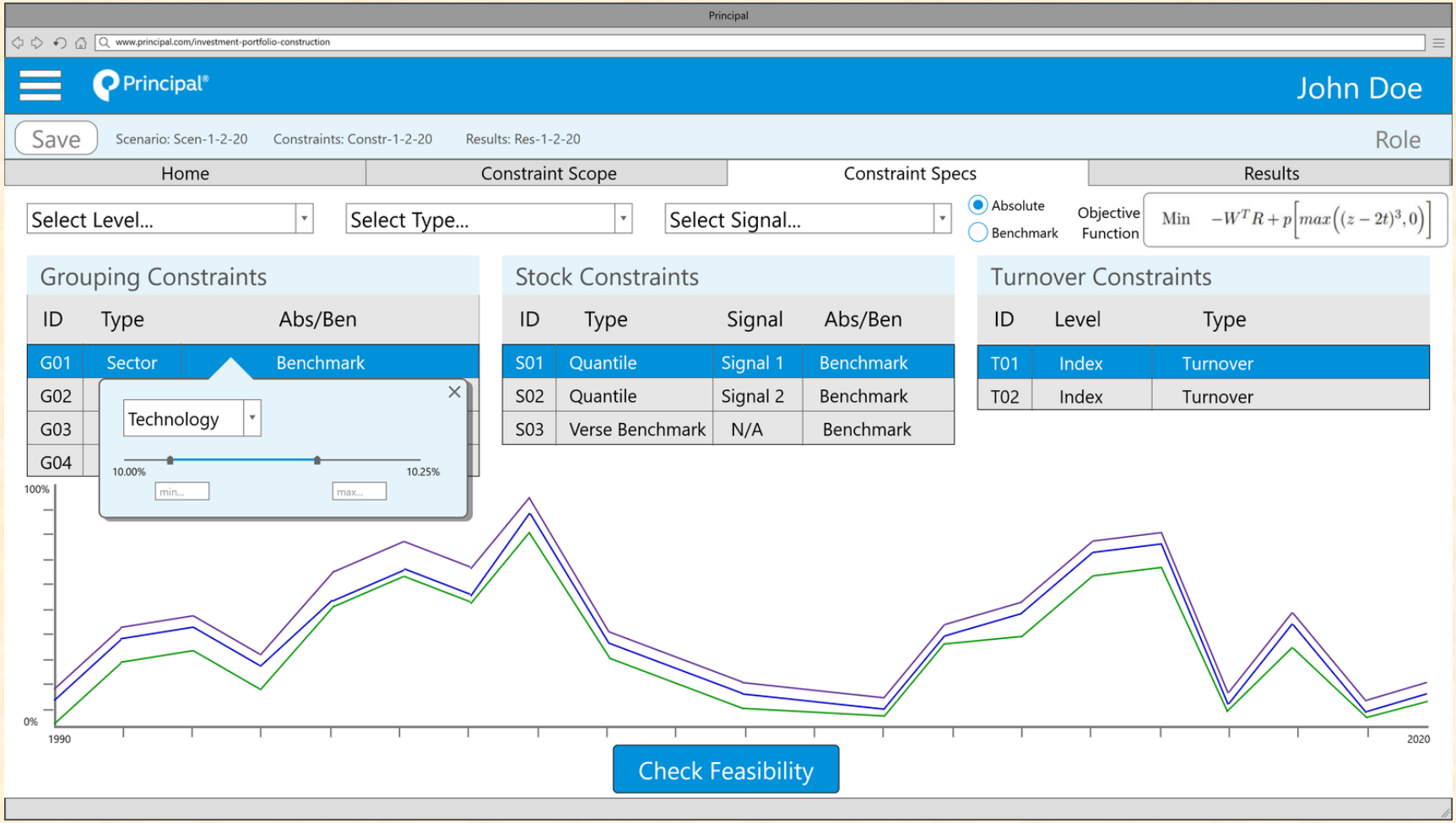
Screen Mockup: Landing/Home Page



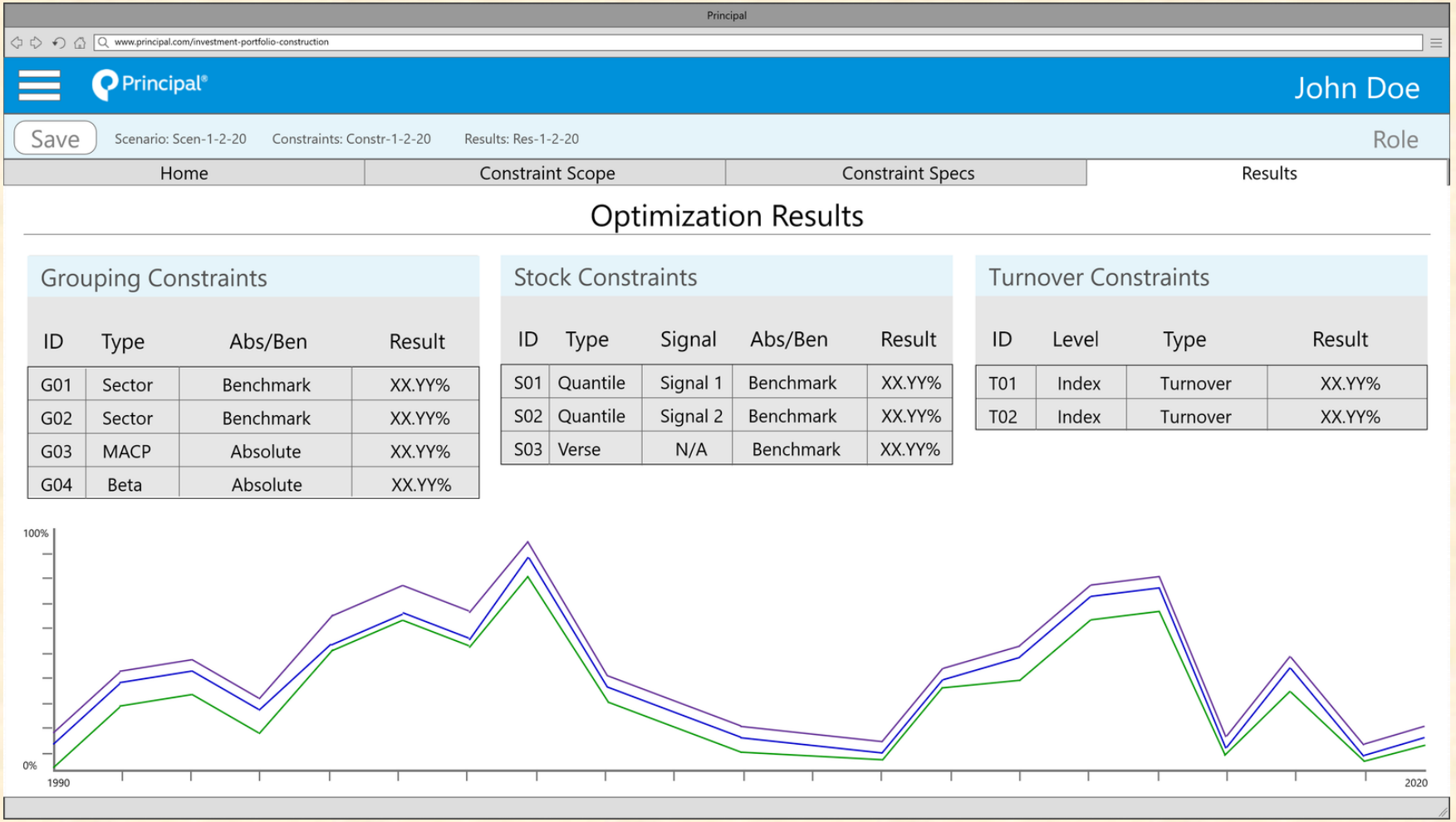
Screen Mockup: Constraint Scoping



Screen Mockup: Constraint Specifications



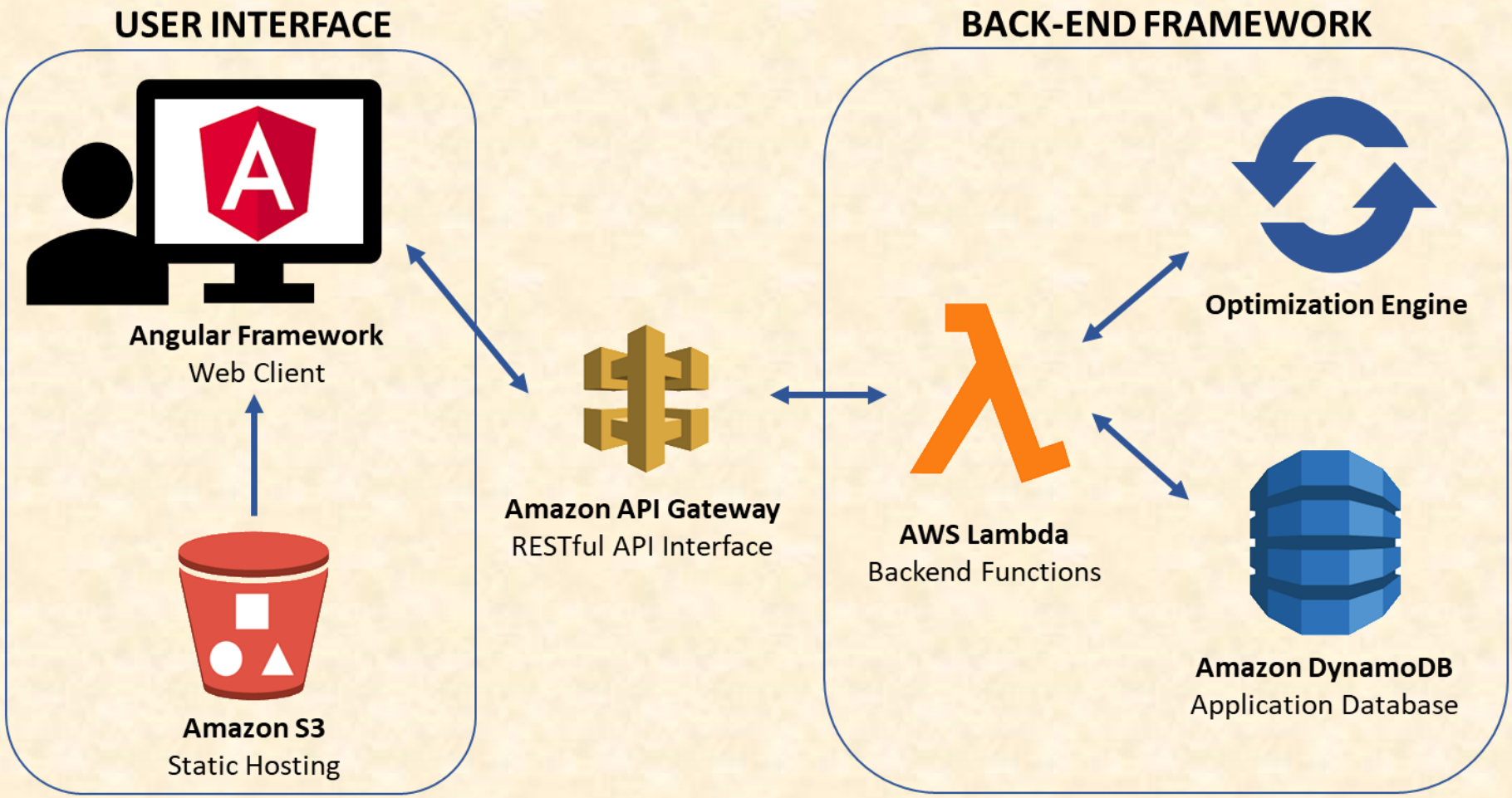
Screen Mockup: Results



Technical Specifications

- **Connection to the optimization engine and database:** a method is required for passing constraints, objective functions, and data into the optimizer
- **Constraint and objective function builder:** programmed equations let users create and store new objective functions and constraints
- **User interface:** users on the front-end can interactively specify constraints that can be saved into databases and loaded for future use

System Architecture



System Components

- Hardware Platforms
  - Computer interface required to use application
- Software Platforms / Technologies
  - Angular (implement user interface)
  - AWS S3 (host Angular application, store and retrieve local data)
  - Amazon API Gateway (link between front-end and back-end)
  - AWS Lambda (provide an environment for combining and returning the results of database queries to the front-end)
  - AWS DynamoDB (store data)

Risks

- **Compatibility with Principal's optimization engine**
  - Method for passing all data to optimization engine must be compatible with current process for accepting data
  - We are working with the client so that the method can be tested early and often directly on their optimizer
- **Serverless Architecture Model (SAM)**
  - Neither system parts nor communication can be dependent on servers
  - We are separating static elements and dynamic elements before implementation
- **API Gateway**
  - All data exchanged between user interface and back-end platforms must be passed through an API gateway designed by the team
  - We are using test functions to ensure gateway can reliably pass mock data between all necessary system components
- **Compatibility with other Principal team's user model**
  - Assignment to worldviews relies on implementation of user groups
  - We are working with Principal to understand the format of user group data

Questions?