CS ADVISING WEBSITE

BRIAN HOOPER, NICK ROHDE, RICO ADRIAN, SHAYLYNN MCDONALD

INTRODUCTION

- The CS department is changing how student advising works
- Currently graduation plans must be manually generated
- Current process is difficult and time-consuming

OUR PROJECT

- Automate the generation of student graduation plans
- Provide an easy to use website for creating student class schedules
- Generate documents that can be used for future advising and registration
- Save time for both advisors and students
- Eliminate errors by automatically verifying graduation plans

FUNCTIONS

- Provides an interface for advisors to generate graduation plans for Computer Science students
- Able to format generated plans for printing
- Contains a database of course offerings and graduation requirements
- Provides an interface for advisors to modify course offerings

DESIGN CONSTRAINTS

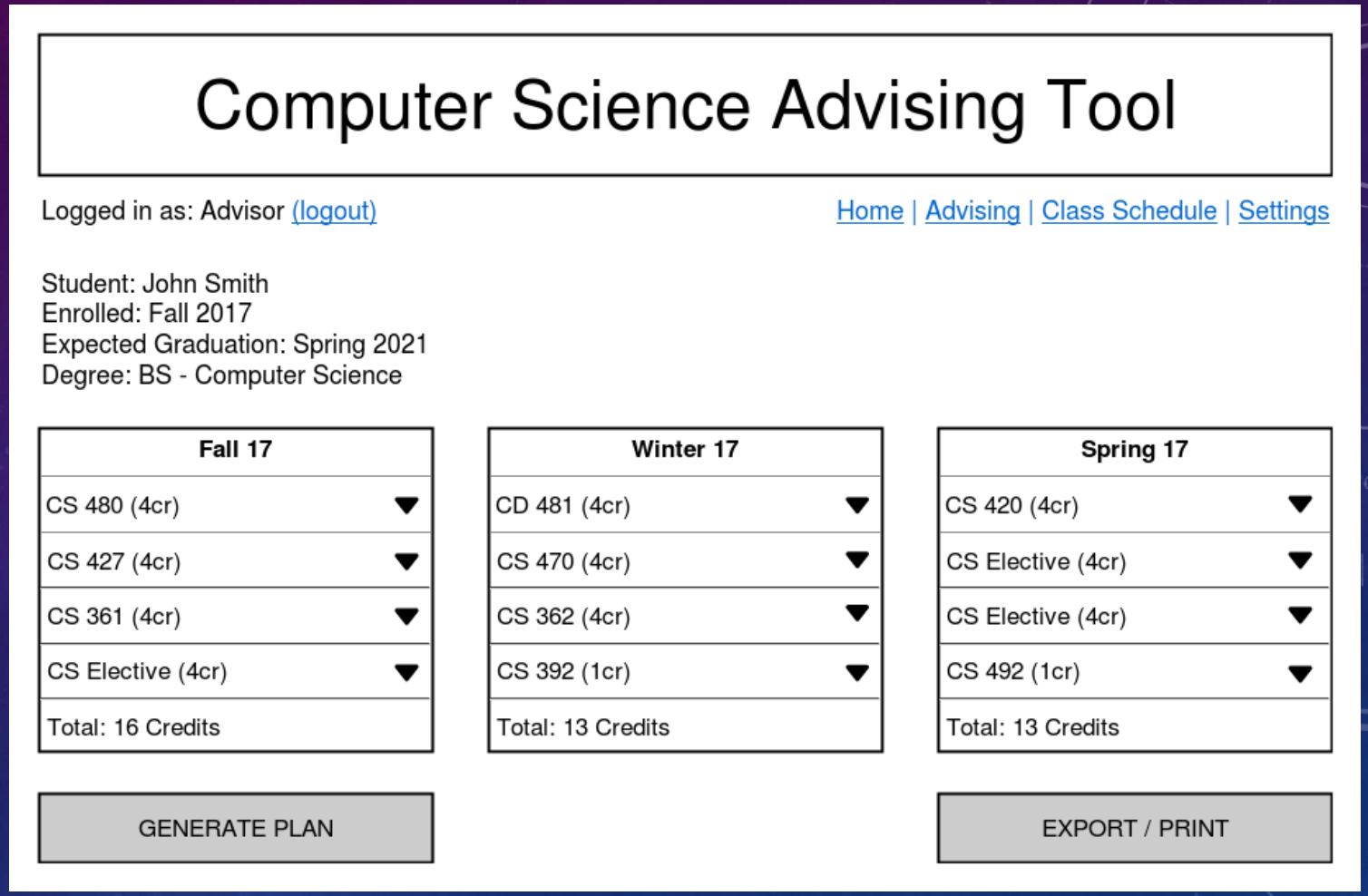
- Prioritizes graduation as early as possible
- Plans include student / instructor class preference, constraints, overridden requirements
- Plans must be verifiable for accuracy, all prerequisites met, quarterly class offerings, etc.
- Privacy / FERPA
- Algorithm speed / Database load

WEBSITE INTERFACE

- Login page
  - Username and Password
- Interface for generating graduation plans
- Interface for modifying course information

WEBSITE UI

- Specific classes chosen using drop down boxes
- Other classes automatically filled in by the scheduling algorithm



ADMINISTRATOR INTERFACE

- Add / Remove course offerings and schedule
- Modify catalog data / graduation requirements
- Import/Modify student information
- Manage user accounts in this product

ADVISOR INTERFACE

- Input student preferences and constraints
- Generate Graduation Plan
- Populate forms

OPERATING ENVIRONMENT

- C# for scheduling algorithm
- Hosted on virtual Linux server
- Modern hardware on server
- Accessed from modern browsers such as Mozilla Firefox 57.0.x

SOFTWARE INTERFACES

- Browser
- Databases
- Linux API

SYSTEM REQUIREMENTS

- User friendly – little to no experience
- Databases – five total
- Application

DEVELOPMENT TOOLS

- Programming Language – C#
- Utilize .NET core for Linux
  - ASP for website
  - SQL/NoSQL tools for databases
- Two database types:
  - MySQL relational database
  - DB4O object database

DATABASES

- SQL Database (MySQL):
  - Graduation Plan – stores plan as a list
  - Login Credentials – stores login credentials
- NoSQL Database (DB4O):
  - Course – stores course objects
  - Catalog – stores catalog objects
  - Student – stores student objects
- Database Handler:
  - Each database has its own handler
  - Handler is middleman between client and database
  - Protect data from multiple writes – each object has write identifier

DATABASES – SQL DATABASE

- One MySQL database with two sub-databases
- Credentials are stored as login (key) and password hash
- Each plan uses student's ID as key
- Quarters are relative to start
- All students have a plan (empty if none was created)

Student ID	Write	Start Qrt	Qrt 1	Qrt 2	...	Qrt 12	...	Qrt K
12345678	7	Fall 17	CS101, CS110	CS111, GEO 120	...	Null	...	Null
22345678	2	Winter 14	CS110, HIST102	GEO120, ENG101	...	CS489, CS470	...	CS492
32345678	1	Sum 15	Null	Null	...	Null	...	Null

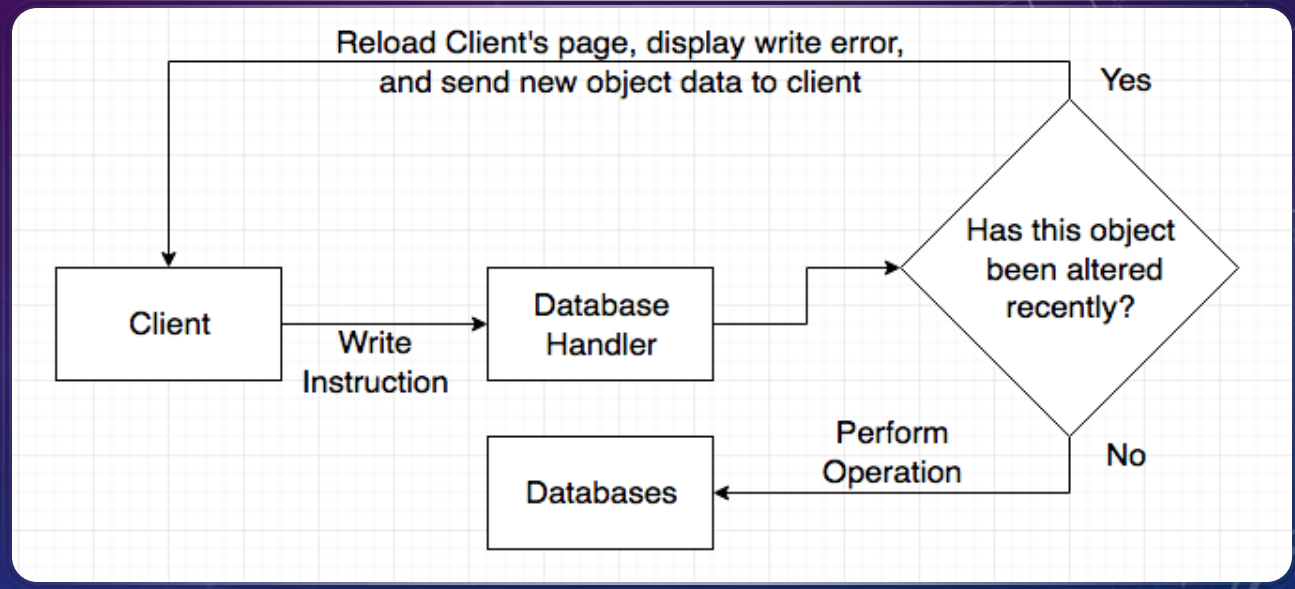
DATABASES – OBJECT DATABASE

- One NoSQL Database with three sub-databases
- Each object has a unique identifier, namely student ID, course ID, or Year
- Allows easy retrieval of data

Object ID	Referenced Object
12345678	Student Object
CS480	Course Object
2015	Catalog Object

DATABASES – HANDLER

- All database operations go from client to Handler
- Operations put into priority queue
  - Admin operations have higher priority
  - Write has priority over read
- Write operations go through a check
  - Each object has write-identifier
- Multiple writes on same object can cause data corruption



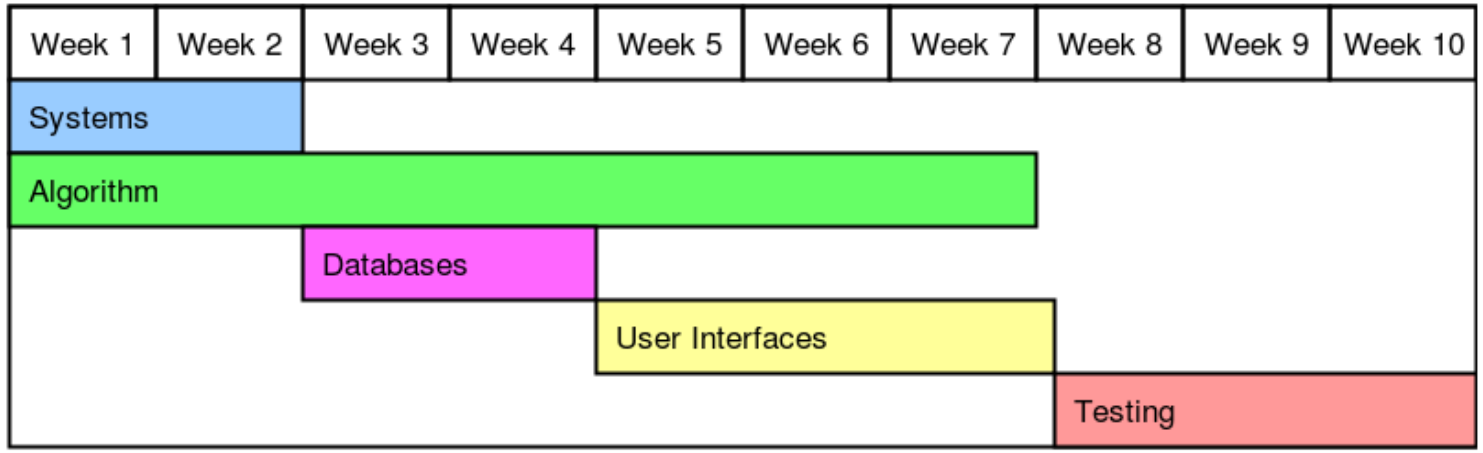
TEST SCENARIOS

- Functional Tests
- Black-Box Testing
- White-Box Testing
- Algorithm accuracy

PROJECT SCHEDULE

- Gantt Chart
- Time approximation
- Project timeline

GANTT CHARTS



Systems design:
Set up application / database server / interfaces

Algorithm implementation:
Program scheduling application

Database set up:
Write database schema and populate with class / catalog information

UI design / implementation:
Design and implement front-end user interface

Testing:
Functionality and UI tests, verify accuracy of algorithm, fix bugs

QUESTIONS