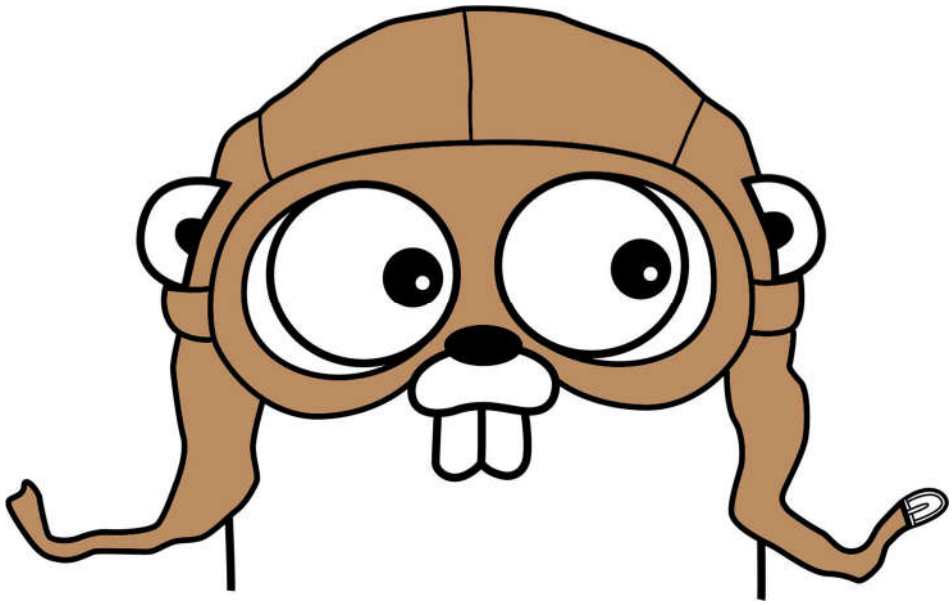
CST8218 – WEB APPLICATION PROGRAMMING

VERSION 0.2 STATUS REPORT

GOPHER



Prepared For:
Professor Douglas King

Prepared By:
Justin Bertrand
Ian Gabriel
Josh Robertson
Skye Turriff
Brandon Yip

Date of Submission:
March 11, 2016

CONTENTS

INTRODUCTION

DESCRIPTION OF THE SYSTEM

The Gopher application allows users to register as either errand runners, known as "Gophers", or errand requesters, known as "Customers" and interact in a structured way to get tasks done. Customers can place orders for an errand that they would like to be done (shopping, pick-up/delivery of packages, etc.) and propose a reward for that errand. A Gopher can then accept and complete the task, for a reward of some kind.

Features may include:

Errand Scheduling: Users are able to schedule errands to be run by Gophers. Errands can be scheduled to be completed at a specific time, or can be requested to be done ASAP. Errands will have a "Reward", which can be of monetary or material value.

Errand Categories: Errands will be able to be tagged into categories, allowing similarly completed tasks to be grouped together.

Gopher Selection: Gophers will be able to commit to completing tasks from the list.

Gopher Review: Users will be able to review their Gopher after the task has been completed, and reviews will be aggregated to determine Gopher quality.

Geolocation System: Users will be able to see relevant Gopher progress geolocated, to determine time of delivery, or time of task completion. This feature is dependant on task type.

LIST OF TEAM MEMBERS

Skye Turriff
Josh Robertson
Justin Bertrand
Ian Gabriel
Brandon Yip

2.0 ARCHITECTURE

2.1 DATA ASPECT

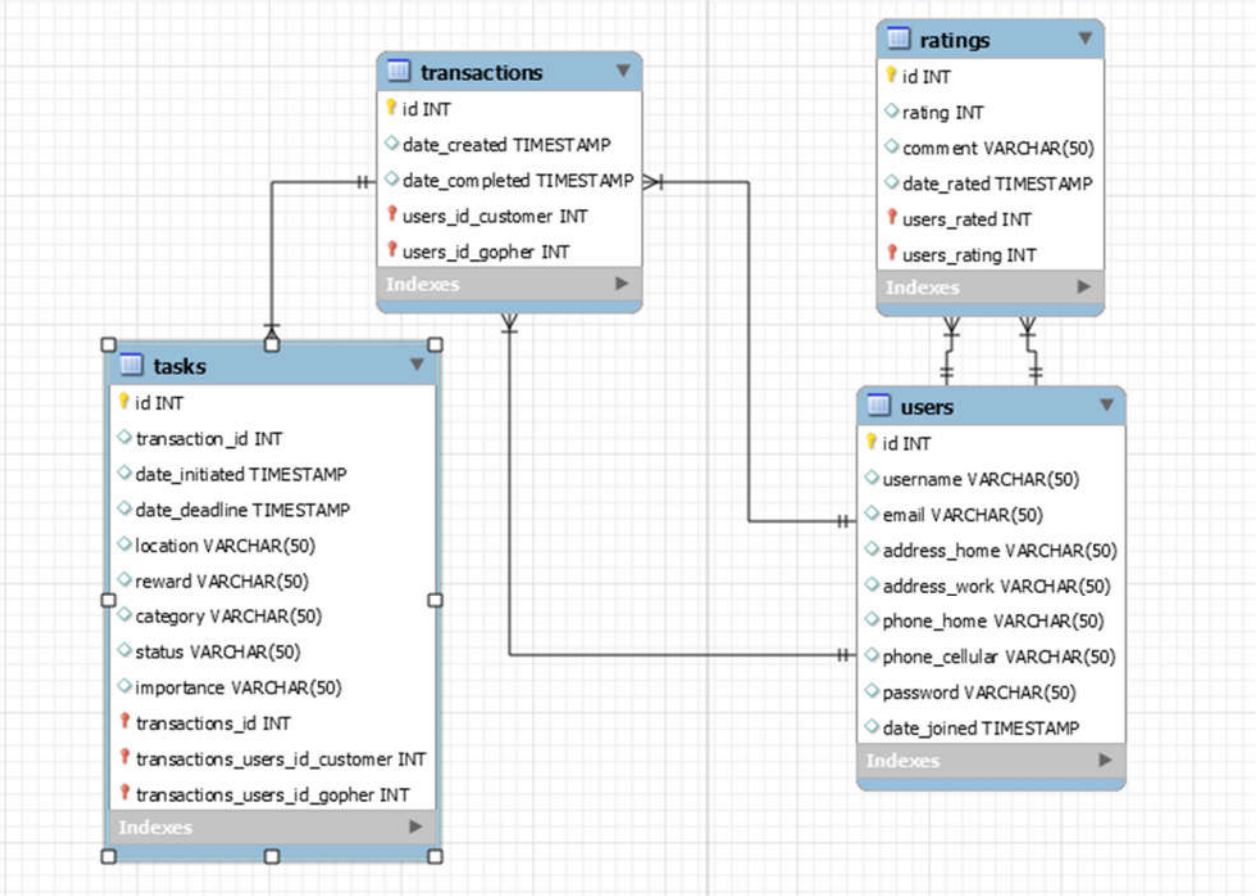


Figure 1: Proposed Entity Relationship Diagram

Very little data will be stored on the client. We will store user authentication information on a session basis for now, with a stretch goal of using cookies to persist user authentication across sessions. We have three types of users, Customers, Gophers, and Moderators/Admins. Transactions will consist of completed Errands and their payments. Errands will consist of task locations, task descriptions, and completion times and requirements.

Note: the following data and relationships are subject to change

Users: Username, email address, home address, work address, phone number, password, date joined

Transactions: Customer, Gopher, Errands, date completed, timestamp of creation

Task: Customer, Gopher, date initiated, description, required completion date and time, location list, reward, errand category, errand status, Transaction id, importance

Ratings: Account being rated, account performing the rating, rating given, comment, timestamp of creation

Data privacy will be a big issue for us. Users should not be able to access the personal details of other users outside of informal cursory information. In a distributed situation the universal source of truth will be the database, as every detail is important to a transaction, and must be shared across connections.

2.2 FUNCTION ASPECT

Errand Requesting: Customers can submit new errand requests to the system using an errand request form. This will include a description of the errand, the pick-up and/or drop-off locations if applicable, a description of the reward offered for completion of the errand, and the time the customer needs the errand completed by.

Viewing and Accepting Errands: Gophers can view unassigned errands in their area, and can accept errands they wish to complete.

Errand Tracking and Notifications: Customers receive updates on the status of their errand, including the location of the assigned Gopher. Updates will be in the form of email notifications, and possibly desktop notifications. As the Gopher progresses through completing the errand, they will be able to modify their status, which will then trigger an update to be sent to the Customer. Examples of status include:

- beginning the errand, in which the errand becomes active and location tracking begins
- (?)errand pick-up in progress, in which the Gopher is on-route to picking up any items required
- errand objective completion, in which any action required (such as picking up or purchasing an item) for the errand has been completed

- (?)errand delivery on-route, in which the Gopher is on his way to delivering any items to the drop-off location requested by the customer
- errand completion, in which the Gopher has successfully completed all errand requirements

Gopher Rating: Customers can rate Gophers after they have completed their errands.

Customer Rating: Gophers can rate customers for whom they have completed errands.

Search Functionality: Users and Gophers will be able to view different collections of errands. Some examples of ways to filter this data are proximity of the relevant locations to the gopher, the to be completed by time, and the errand initialization time.

Administrative Tasks:

- When a Gopher's rating falls below the minimum, their account is put on probation
- Delete illegal or inappropriate errand requests
- Handle disputes, and investigate any incidents of scamming/fraud

2.3 NETWORK ASPECT

Data will predominantly reside in the database. Clients will hold data during sessions.

Users will be accessing the application over the internet, through any type of browser. This could possibly extend to a mobile application in future iterations.

2.4 PEOPLE ASPECT

User Type: Gopher

Users can register as Gophers and have access to view and accept errand requests.

User Type: Customer

Users can register as Customers to post errand requests

Administration Roles:

Moderators to ensure that the errands posted on the message board adhere to the Gopher Guidelines.

2.5 TIMING ASPECT

Account Creation
Customer and Gopher accounts are created immediately, without requiring authorization from an administrator.

Errand Request Submission
When a customer submits an errand request, it is posted and immediately viewable by Gophers.

Errand Acceptance
When a Gopher accepts an errand request, the request is immediately added to the Gopher's active errands

Errand Time Constraints
Errands may have a time constraint, and will expire if not accepted by a Gopher within 24 hours of a task being due. Other tasks will not have a time constraint.

2.6 MOTIVATION ASPECT

Privacy Issues

User passwords will need to be encrypted. Errand information will need to be private to the requester and the assigned runner.

Location information will need to be kept private since the provided locations for errands can often be work or home addresses. Gopher's locations will only be available to the User for the duration of an errand.

Compliance & Regulatory Issues

- Payment Card Industry Data Security Standard compliancy
- Errands must comply with law and regulations of the area they are conducted in
- Runners must maintain a certain rating to keep running errands
- Personal Information Protection and Electronic Documents Act (PIPEDA)

3.0 PROJECT PLAN

3.1 MAPPING OF FEATURES

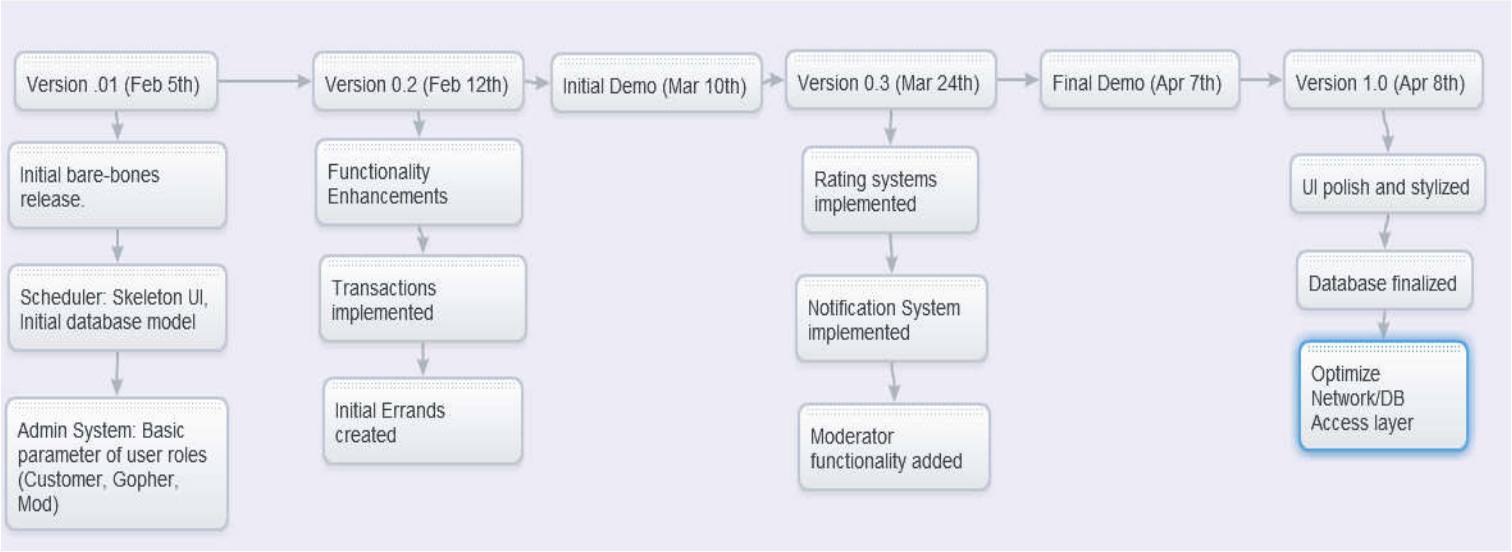


Figure 2: Mapping of Features and Components

Figure 2 above represents the proposed workflow for Gopher application development.

3.2 TASK BREAKDOWN

****Subject to Change**

Table 1: Task Breakdown

Member	Justin	Josh	Ian	Skye	Brandon
Version 0.1	Database design	Layout for errand request form	Layout for login, register, and forgot password pages	Layout for Gopher view of errands	Layout for home screen/ welcome page, invalid login page
Version 0.2	International ization of components	deployment of database	implement transactions	business logic for managing errands	account authorization /authenticati on
Version 0.3	implement rating system for Gophers and Customers	implement Gopher's ability to change status of current errand	integrate location tracking	implement notifications	add moderator functionality
Version 1.0	polish UI	optimize app access to database	finalize database	optimize app access to database/per formance factors	optimize business logic

Table 2: Feature Implementation Schedule

Component	Version 0.1	Version 0.2	Version 0.3	Version 1.0
User Interface	All	All	All	All
Database	All	Josh	All	All
Serverside	Not Implemented	All	Josh, Skye, Brandon	All
Rating System	Not Implemented	Not Implemented	Justin	All
Geolocation	Not Implemented	Not Implemented	Ian	All

4.0 STATUS REPORT

After the development of version 0.1, we had a mostly complete user interface for the Gopher web application. This gave an us excellent overview of how the application would flow between web pages, where specific data would need to be accessed and displayed, as well as how that data would persist and travel throughout the application. These insights heavily influenced our development process for version 0.2.

The version 0.1 database structure was further flushed out and expanded. The new implementation for version 0.2 establishes clearer relationships between users and errands, errands and tasks, tasks and locations, users and locations. Each table was further normalized and designed to act as a single area of concern and better map to the application data objects, with foreign key references to establish relations to other data objects. Look up tables were added in order to provide a standard to categorize or index tables that have the potential to grow quite large, such as the addresses or errands table. SQL scripts were also created to populate the database with computer generated data in order to begin accessing and displaying that data in the Gopher web application.

Much of the time spent for version 0.2 was in the form of research and planning. A lot of effort was required to research JSP, DAO, and Servlet technology, mainly:

- How these technologies are supposed to interact and function in a web application,
- Proper coding standards and current design pattern implementations for these technologies,
- How data is marshalled into the web application using these technologies,
- How user authentication and authorization is implemented,
- How sensitive data should be encrypted, transferred, and stored,
- How JSTL is used in JSPs to display data, etc.

Because of the unanticipated effort that was required outside of class time to better understand the above Java EE technologies and how they are implemented in code, the goal for the end of version 0.2 was updated to have the Gopher web application composed of a View layer of JSPs, a Data

Access Layer composed of DAO classes and POJOs, and a Controller layer composed of Servlet classes. This differed from the original project plan for version 0.2, which included internationalization and processing transactions. These two components were not included in the release of version 0.2, and will be moved forward to be included in version 0.3.

4.1 UPDATED ARCHITECTURE: DATA ASPECT

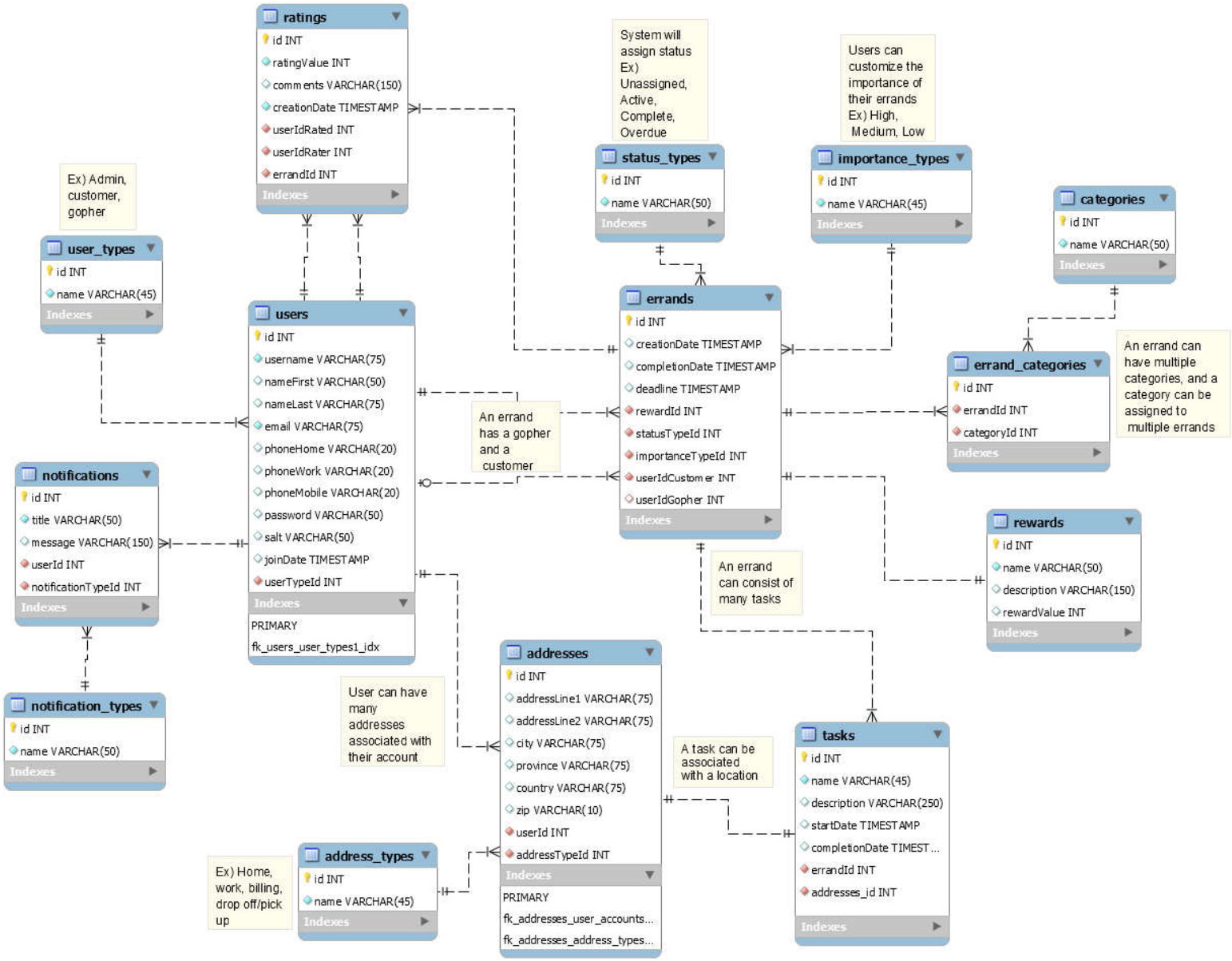


Figure 3: Updated Entity Relationship Diagram

4.2 UPDATED PROJECT PLAN: TASK BREAKDOWN

Table III: Updated Task Breakdown

Member	Justin	Josh	Ian	Skye	Brandon
Version 0.1	Database design	Layout for errand request form	Layout for login, register, and forgot password pages	Layout for Gopher view of errands	Layout for home screen/ welcome page, invalid login page
Version 0.2	Password encryption and storage, user login and authentication and registration, deployment of database on hosting service and configuration, Servlet and DAO classes	Browse/Errand JSPs and Servlet classes, DAO classes, organize project structure, JSTL implementation, data generation and SQL insert script for database version 0.2	Refactoring all entities classes to map to database version 0.2 fields, generating data for database version 0.2, SQL insert script, Servlet and DOA classes	Database version 0.2 design and creation, SQL creation script, Dashboard JSP, Servlet, and DAO, Rating DAO, updating User entity and DAO	User interface design and components, web app logo, creation of common look and feel for web application

Table IV: Updated Feature Implementation Schedule

Component	Version 0.1	Version 0.2	Version 0.3	Version 1.0
User Interface	All	All	All	All
Database	All	All	All	All
Server-side	Not Implemented	All	All	All
Rating System	Not Implemented	Not Implemented	All	All
Geolocation	Not Implemented	Not Implemented	All	All