Instructions for team selection

Milestone 2

Please register your team here.

Next slide is from ECE 297 students for ECE 297 students

Sustainable Engineers Association

4th ANNUAL

SUSTAINABILITY CONFERENCE

Sat. Feb 1st, 2014

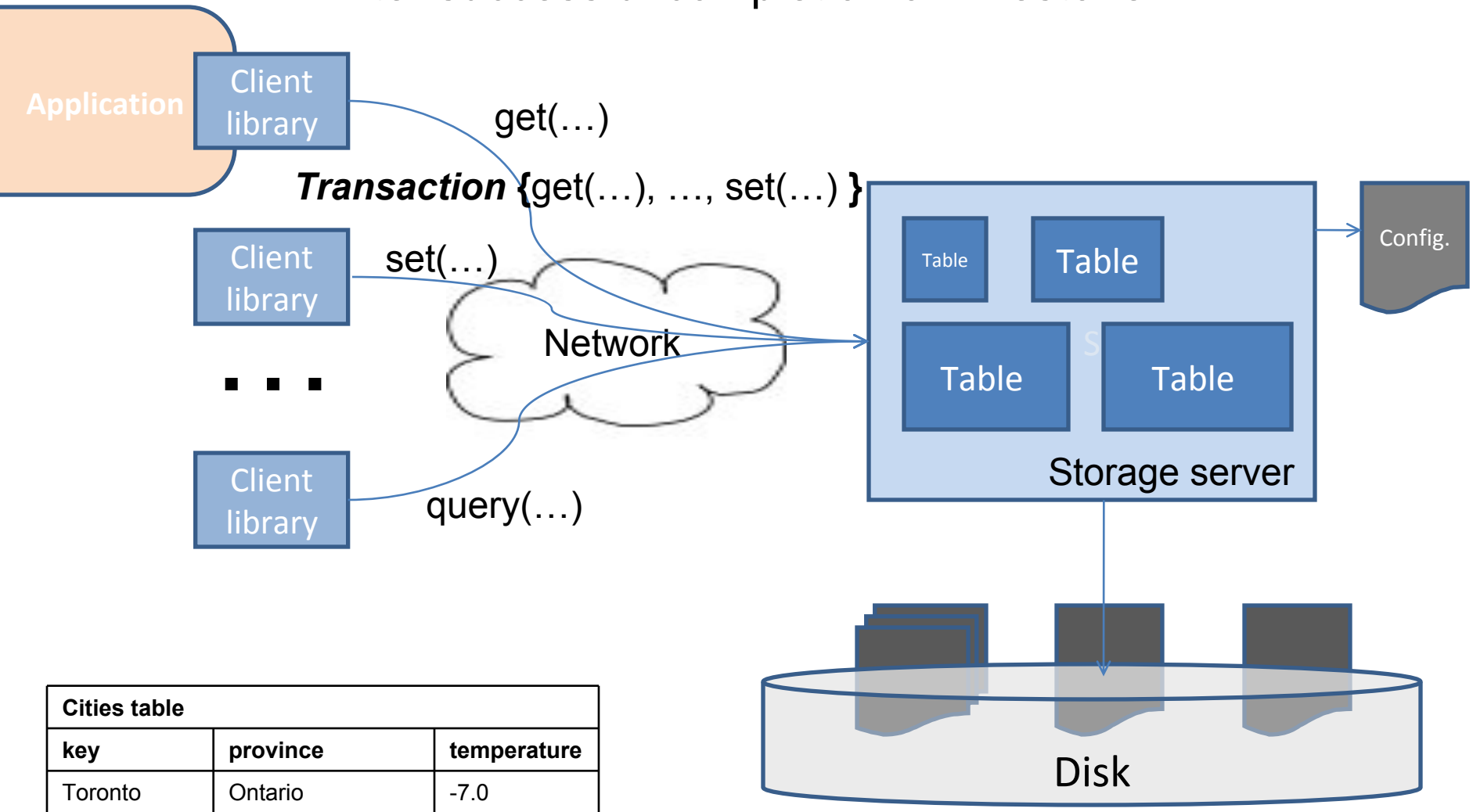
Bahen Centre for Information Technology, 9:30am - 6:00pm

HIGHLIGHTING:
Transition to a Sustainable Energy Future and Building Sustainable Cities

FOR MORE INFORMATION AND REGISTRATION
http://www.sustainable-engineers.org/sea-conference-2014/

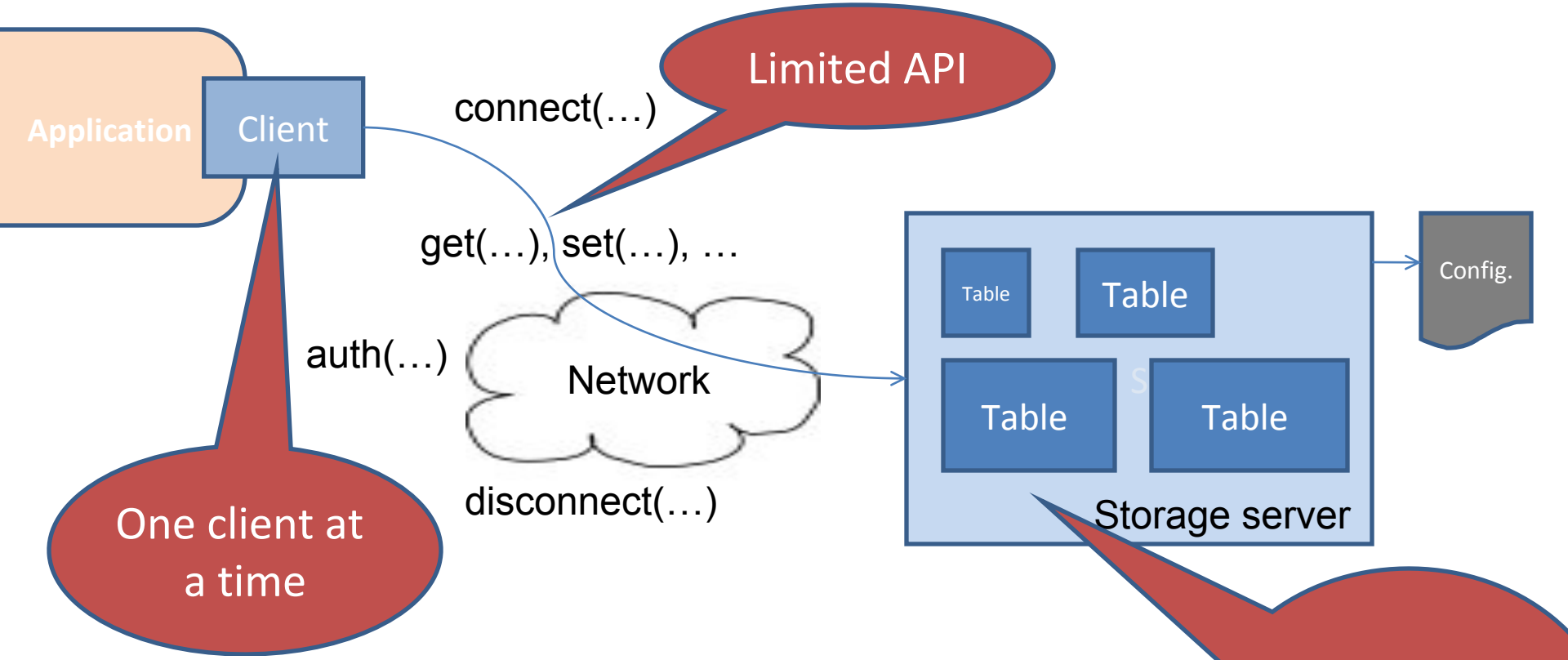
The final storage server

After successful completion of Milestone 4



Cities table		
key	**province**	**temperature**
Toronto	Ontario	-7.0
Winnipeg	Manitoba	-17.4
Vancouver	British Columbia	8.1

Storage server Milestone 2



Citie		
key		
Toro		
Winnipeg	Manitoba	-17.4
Vancouver	British Columbia	8.1

"Toronto Ontario -7.0"

Data schema is strings only

Tables and records

- Storage server manages ***tables***
- A table is a collection of ***records***
- Each record is **uniquely** identified by a ***key***
- Records comprise *one or more* ***values***
- A **record** is a string with a maximum number of characters (Milestone 2!)

Course		
ECE297	**"95 Aplus"**	
ECE344		
ECE451	87	A

Keys are unique

Course	Mark
ECE297	95
ECE344	92
ECE451	87

Course	Mark
ECE297	95
ECE344	92
ECE297	87
ECE451	87

Course is the table's key field.
The key must be unique.

Storage server API

void ***storage_connect**(char ***hostname**, int **port**)

int **storage_disconnect**(void ***conn**)

int **storage_auth**(char ***username**, char ***passwd**, void *conn)

int **storage_get**(char ***table**, char ***key**, struct storage_record ***record**, void ***conn**)

int **storage_set**(char ***table**, char ***key**, struct storage_record ***record**, void ***conn**)

What do I need to implement?

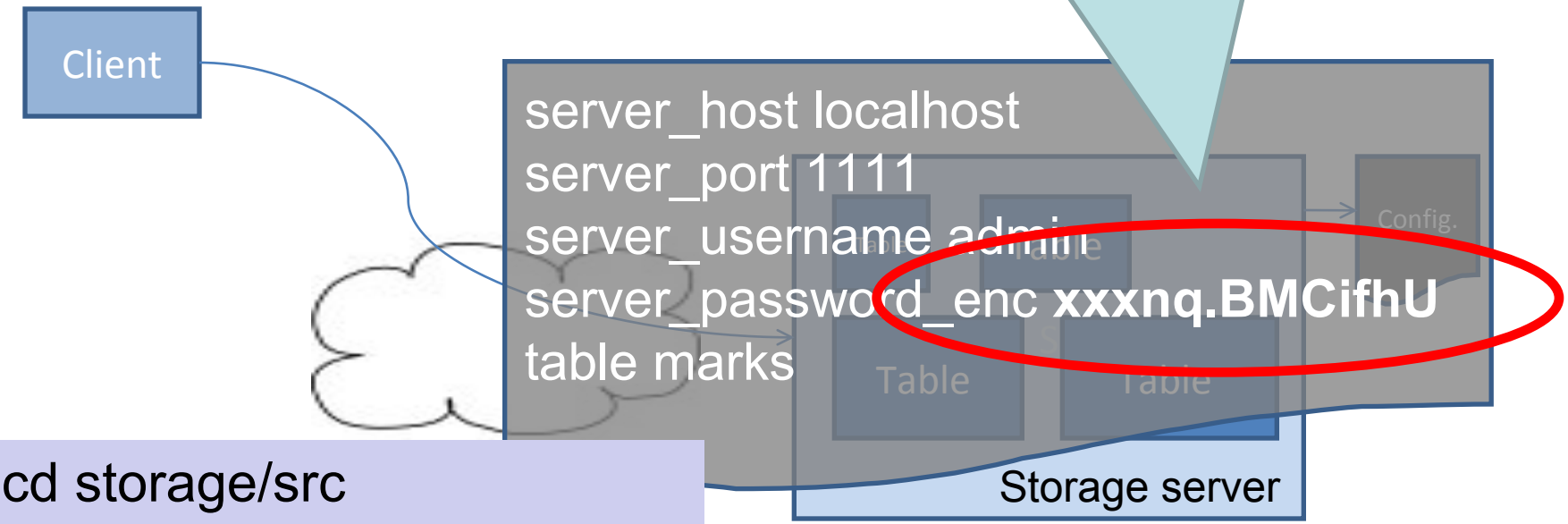
What do I need to implement?

- Client-side library functions (get, set etc.)
- Server-side functions (get, set etc.)
- Communication between client and server
  - *How does the server know what function is called?*
  - *How does the server know what the input to the function is?*
- Table management
  - *How can the server find a table, a record in a table? Determine the table/record does not exist?*

What do I need to implement?

- Client-server authentication
  - … storage_connect(…)
  - … storage_auth("***ad***
  - …
  - … storage_disconnect(…)

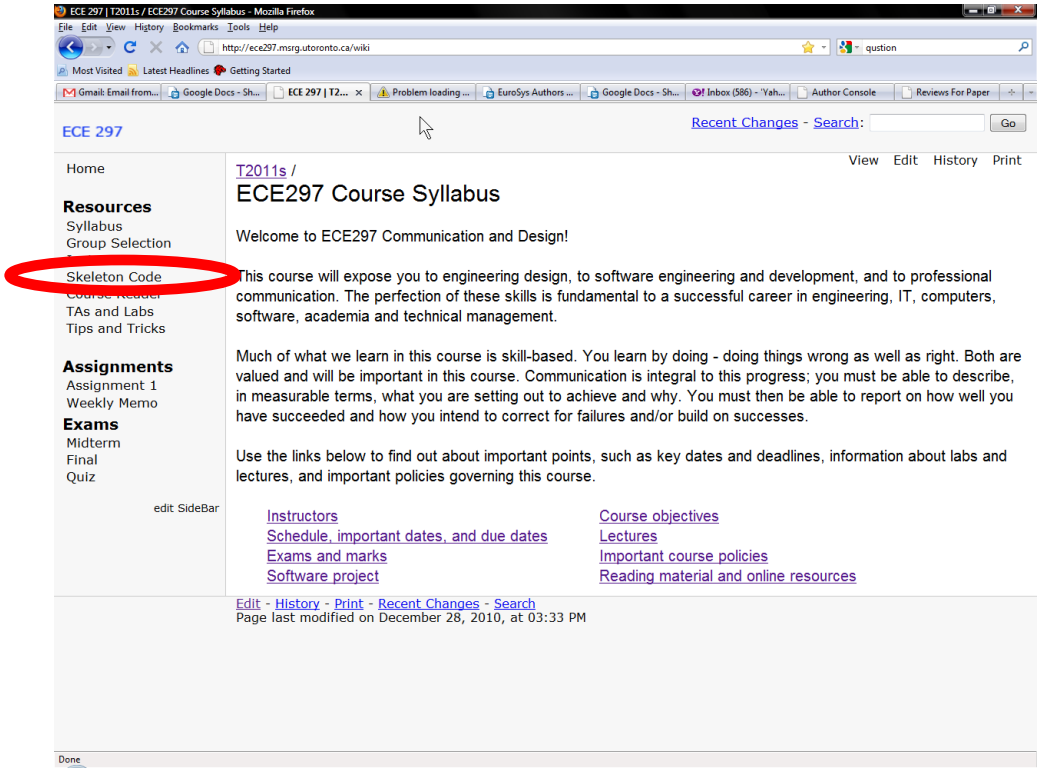
Authentication on the server side & associated error handling is missing




```
> cd storage/src
> make encrypt_passwd
> ./encrypt_passwd dog4sale
  xxxnq.BMCifhU
```

What don't I need to implement?

- Read the skeleton code!



get & set

- int storage_get(
  const char *table,
  const char *key,
  struct storage_record *record,
  void *conn
  );
- int storage_set(
  const char *table, const char *key,
  struct storage_record *record,
  void *conn
  );

??

?

"MyCourses"

Course	Mark
ECE297	95
ECE344	92
ECE451	87

```
struct storage_record {
        char value[MAX_VALUE_LEN];
        uintptr_t metadata[8];
};
```
(***given, see handout***)

???

What else do we need?

- Besides retrieving (get) and inserting (set) a record?

Functionality

- **Retrieve** an existing record
  - storage_get(…) function
- **Insert** a new record
  - storage_set(…) with a **key** that **does not already exist** in the table
- **Update** an **existing record**
  - storage_set(..) with a **key** that **already exists** in the table
- **Delete** a record
  - storage_set() with a **key** that **already exists** in the table and **a NULL value** as record

Error conditions

- ERR_INVALID_PARAM (all 5 functions)
  - parameters do not conform to the specification.
- ERR_CONNECTION_FAIL (all 5 functions)
  - connection problems to server
- ERR_AUTHENTICATION_FAILED
  - client-server authentication problems
- ERR_TABLE_NOT_FOUND (storage_get()/_set)
  - specified table does not exist
- ERR_KEY_NOT_FOUND (storage_get())
  - server indicates that the specified key does not exist in the specified table.
- ERR_UNKNOWN
  - flag any other errors (out of memory, file not found, …)

Let’s design storage_get(…)

//storage_get(*table, *key, *storage_record, *conn)

storage_get(…) specification

//storage_get(*table, *key, *storage_record, *conn)

(Copied from handout)

- **table**: A table in the database. **key**: A key in the table. **record**: A pointer to a record structure; …
- The record with the specified key in the specified table is retrieved from the server using the specified connection.
  - If the key is found, the record structure is populated with the details of the corresponding record.
  - Otherwise, the record structure is not modified.
- Return 0 if successful, and -1 otherwise.
- Need to think about client pieces and server pieces

Where do I start?

At this point interactions with TAs useful

- Read the handout for Milestone 2
  - Play with svn *et al.* (see handout)
- Register team (note our deadline)
- Brainstorm on the overall design as a team
- Divide into tasks (see our suggestions)
- Do some research on your assigned tasks (see pointers in Course Reader)
- Do a design for your task
- meet Meet with your team and discuss
- meet Keep design notes (e.g., G. Docs)
- Implement some, test some, …
- meet

■ ■ ■

Make it a habit of meeting for 15 minutes every other day (e.g., phone) to report on status.

“Scrum in under 10 minutes”

- Adopt an agile development process
- Find and watch the video online
  - “*Scrum in under 10 minutes*”

Design considerations

- How should the data (i.e., tables, records, keys) managed by the storage server be stored in memory?

Design considerations

- How do the server and client library handle failures?
  - What happens if there's an error in the configuration file?
  - What if the server is not running?
  - What if an invalid table name is given?

Questions?