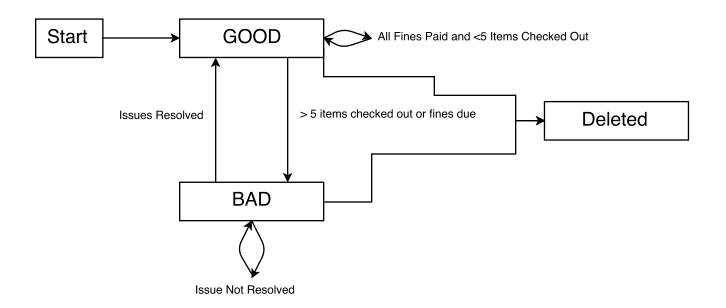
Justin Wei	JustinWei@my.unt.edu	CSCE1040		Name	Туре	Class	Description
				title	string	DVD	title
Name	Туре	Class	Description	production st	tudio string	DVD	production studio
id count	int	patrons	used for generating unique ids	duration	time_t	DVD	duration
roster	vector <patron></patron>	patrons	holds patrons				
				Name	Туре	Class	Description
Name	Туре	Class	Description	artist	string	CD	artist
patron id	int	patron	patron id	label	string	CD	label
num books	int	patron	num books borrowed				
num avs	int	patron	num avs borroed	Name	Туре	Class	Description
name	string	patron	name	title	string	Book	title
status	string	patron	status	author	string	Book	author
address	string	patron	address	publisher	string	Book	publisher
phone	string	patron	phone number				
fine	float	patron	fine balance	Name	Туре	Class	Description
				title	string	DVD	title
Name	Туре	Class	Description	author	string	DVD	author
id count	int	libitems	used for generating unique ids	publisher	string	DVD	publisher
items	vector <libitem*></libitem*>	libitems	holds libitem pointers	editor	string	DVD	editor
Name	Туре	Class	Description	Name	Туре	Class	Description
id	string	libitem	libitem id	id	int	Transaction	id
status	string	libitem	status	patronid	int	Transaction	patronid
type	string	libitem	type of item	item id	int	Transaction	item id
loan period	time_t	libitem	loadn period	due	time_t	Transaction	due
fine rate	float	libitem	fine rate	renewable	bool	Transaction	renewable
replacement cost	float	libitem	replacement cost				
renewable	bool	libitem	renewable	Name	Туре	Class	Description
borrowable	bool	libitem	borrowable	idcount	int	Transactions	used for generating unique ids
				list	vecotr <transaction></transaction>	Transactions	holds transactions
				ps	patrons*	Transactions	patrons contrioller
				Is	libitems{*	Transactions	libitems controller

**Transactions Patrons** LibItems private: private: int id\_count; private: int id count; vector < transaction > list; vector < patron > roster; int id count; patrons \* ps; vector < libitem \* > items; libitems \* ls; public: public: public: //Constructors //Constructors patrons(): //Constructors libitems(); ~patrons(); transactions(patrons \* , libitems \* ); ~libitems(); ~transactions(); int get\_id\_count(); //Getters string generate\_id(); int get\_id\_count(); //Getters void print(); int get\_count(); int get\_id\_count(); void add(libitem \* ); void remove(); //Print //Transaction Functions void edit(); void print\_mailing(); void lose(); libitem \* find(string); void print(); void check\_out(); bool in (string id); void print\_fines(); void remove(); void admin\_edit(); void remove(int id): //Patron Functions void add(transaction); void load libitems(ifstream & ); void add(); void renew(); void load\_books(ifstream & ); void add(patron); void renew(int); void load\_CDs(ifstream & ); void remove(); void load\_DVDs(ifstream & ); void edit(); //Print void load\_references(ifstream & ); void pay\_fine(); void print(): void print\_patrons\_with\_overdue(); void save libitems(FILE \* ): //find void print\_patron\_checked\_out(); void save books(FILE \* ); patron & find(int); void print\_overdue\_items(); void save\_CDs(FILE \* ); bool in (int id); void save\_DVDs(FILE \* ); //Find / In void save\_references(FILE \* ); //Save / Load bool in (int id): void save\_patrons(FILE \* ); transaction & find(int id); friend ostream & operator << (ostream void load\_patrons(ifstream & ); & output, libitems \* ls); //Save / Load friend istream & operator >> (istream & //Overloads void load\_transactions(ifstream & ); input, libitems \* ls); friend ostream & operator << void save transactions(FILE \* ); (ostream & output, patrons \* ps); friend istream & operator >> (istream //Checks & , patrons \* ps); int check\_patron\_has\_overdue(int); Manages //Overloads friend ostream & operator << Manages (One to many) (ostream & output, transactions \* ts); friend istream & operator << (istream Patron & input, transactions \* ts); Libltem private: Manages int patron\_id, num\_books, num\_avs; protected: (One to Many) string name, status, address, phone; string id, status, type; float fine; time\_t loan\_period; float fine\_rate, Transaction public: replacement cost; bool renewable, borrowable; private: //Constructors int id, patron\_id; patron(int); public: string item\_id; patron(); Checks Out time\_t due; //Constructors (One to Many) bool renewable; //Add / Pay Fines libitem(); void add fine(float); //print public: void pay\_fine(float); virtual void print(); //Constructors //Print virtual void edit(); transaction(); void print(); transaction(int); Holds //Overload //Checks friend ostream & operator << //print bool check\_fine(); (ostream & , libitem \* ); void print(); bool check\_limit(string type); friend istream & operator >> (istream & , libitem \* ); //Overloads //Overloads friend ostream & operator << friend ostream & operator << (ostream & , transaction); (ostream & , patron); friend istream & operator >> friend istream & operator >> (istream (istream & , transaction); & , patron & );

## LibItem protected: string id, status, type; time\_t loan\_period; float fine rate, replacement\_cost; bool renewable, borrowable; public: //Constructors libitem(); //print virtual void print(); virtual void edit(); //Overload friend ostream & operator << (ostream & , libitem \* ); friend istream & operator >> (istream & , libitem \* ); is derived from Reference Book DVD private: string title, private: string title, author, private: string title, author, editor. production\_studio; publisher; publisher; int duration; CD public: public: private: string artist, public: label; //Constructors //Constructors //Constructors book(); reference(); DVD(); public: book(string); reference(string); DVD(string); //Print //Constructors void print(); void print(); void print(); CD(); void edit(); void edit(); CD(string); //Fuctions friend istream & operator >> private: string artist, friend istream & operator >> void edit(); (istream & , DVD \* ); label; (istream & o, book \* b); //Overloads public: friend istream & operator >> (istream & , reference \* ); //Constructors CD(); CD(string);

## **Patron Status State Diagram**



## **Book Status State Diagram**

